AGENDA

NVIDIA Video Technologies Overview
Turing Video Enhancements
Video Codec SDK Updates
Benchmarks
Roadmap
NVIDIA VIDEO TECHNOLOGIES
NVIDIA GPU VIDEO CAPABILITIES

Decode HW*
Formats:
• MPEG-2
• VC1
• VP8
• VP9
• H.264
• H.265
• Lossless

Bit depth:
• 8/10/12 bit

Color**
• YUV 4:2:0
• YUV 4:4:4

Resolution
• Up to 8K***

* See support diagram for previous NVIDIA HW generations
** 4:4:4 is supported only on HEVC for Turing; 4:2:2 is not natively supported on HW
*** Support is codec dependent

Encode HW*
Formats:
• H.264
• H.265
• Lossless

Bit depth:
• 8 bit
• 10 bit

Color**
• YUV 4:4:4
• YUV 4:2:0

Resolution
• Up to 8K***
VIDEO CODEC SDK
A comprehensive set of APIs for GPU-accelerated video encode and decode

NVENCODE API for video encode acceleration

NVDECODE API for video & JPEG decode acceleration (formerly called NVCUVID API)

Independent of CUDA/3D cores on GPU for pre-/post-processing

Gamestream

Video transcoding

Remote desktop streaming

Intelligent video analytics

Video archiving

Video editing
NVIDIA VIDEO TECHNOLOGIES

SOFTWARE

VIDEO CODEC, OPTICAL FLOW SDK
Video Encode and Decode for Windows and Linux CUDA, DirectX, OpenGL interoperability

CUDA TOOLKIT
APIs, libraries, tools, samples

NVENC
Video encode

NVDEC
Video decode

H.264
MPEG-4/AVC
H.265
HEVC

MPEG2
H.264
MPEG-4/AVC
H.265
HEVC
VP8
VP9

NVIDIA DRIVER

DALI

cuDNN, TensorRT, cuBLAS, cuSPARSE

CUDA
High-performance computing on GPU

Easy access to GPU video acceleration

DeepStream SDK

H.264
MPEG-4/AVC
H.265
HEVC

FFmpeg

H.264
MPEG-4/AVC
H.265
HEVC

H.264
MPEG-4/AVC
H.265
HEVC
VP8
VP9

H.264
MPEG-4/AVC
H.265
HEVC
VIDEO CODEC SDK UPDATE
VIDEO CODEC SDK UPDATE

**SDK 7.x**
- Pascal
- 10-bit encode
- FFmpeg
- ME-only for VR
- Quality++

**SDK 8.0**
- 10-bit transcode
- 10/12-bit decode
- OpenGL
- Dec. optimizations
- WP, AQ, Enc.
- Quality

**SDK 8.1**
- B-as-ref
- QP/emphasis map
- 4K60 HEVC encode
- Reusable classes & new sample apps

**SDK 8.2**
- Decode + inference optimizations

**SDK 8.3**
- Multi-NVDEC
- HEVC 4:4:4 decode
- Encode quality++
- HEVC B frames

**SDK 9.0**
- Turing
- Multi-NVDEC

**Timeline**
- 2016
- 2017
- Q1 2018
- Q3 2018
- 2019
## VIDEO CODEC SDK 9.0

**Soul**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Who it benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Higher video encode quality</td>
<td>Cloud gaming</td>
</tr>
<tr>
<td>HEVC B-frames</td>
<td>Game broadcasting (e.g. Twitch)</td>
</tr>
<tr>
<td>Higher encode quality</td>
<td>Video transcoding (e.g. Youtube, Facebook)</td>
</tr>
<tr>
<td></td>
<td>OTT/M&amp;E</td>
</tr>
<tr>
<td>HEVC 4:4:4 decode</td>
<td>End-to-end high-quality remote desktop</td>
</tr>
<tr>
<td>Mutiple NVDECs</td>
<td>Higher decode + inference throughput</td>
</tr>
<tr>
<td>Direct output to vidmem</td>
<td>Higher perf with post-processing</td>
</tr>
<tr>
<td>Power 9 + Tesla V100 SXM2</td>
<td>Video SDK for IBM platforms</td>
</tr>
</tbody>
</table>
TURING UPDATES - NVDEC
# MULTIPLE NVDECs IN TURING

<table>
<thead>
<tr>
<th>GPU</th>
<th>Number of NVDECs per GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volta, Pascal &amp; earlier</td>
<td>1</td>
</tr>
<tr>
<td>Turing - GeForce (RTX)</td>
<td>1</td>
</tr>
<tr>
<td>Turing - Quadro &amp; Tesla (TU106)</td>
<td>3</td>
</tr>
<tr>
<td>Turing - Quadro &amp; Tesla (TU104)</td>
<td>2</td>
</tr>
<tr>
<td>Turing - others</td>
<td>1</td>
</tr>
</tbody>
</table>

- Quadro & Tesla feature
- Auto-load-balanced by driver
PASCAL & EARLIER

Single NVDEC

Bottleneck

NVDEC

High-res Decode
1080p, 720p

Scale → Infer

Scale → Infer

Scale → Infer

Scale → Infer

Low-res infer e.g. 300 × 200
TURING
Multiple NVDECs

NVDEC 0

NVDEC N

High-res Decode 1080p, 720p

Scale → Infer

Scale → Infer

Scale → Infer

Scale → Infer

Low-res infer e.g. 300 × 200
END-TO-END 4:4:4 IN TURING

➢ Preserves chroma: text and thin lines
➢ Valuable in desktop streaming

4:2:0

4:4:4
END-TO-END 4:4:4 IN TURING
HEVC 4:4:4 HW encode & 4:4:4 HW decode

Desktop Capture → HW Encode → Stream

Network

HW decode → Render
TURING NVENC ENHANCEMENTS
**NVENC - ENCODING QUALITY**

Focus for Turing NVENC

<table>
<thead>
<tr>
<th>Enhancement</th>
<th>How to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rate distortion optimization - RDO</td>
<td>Turing only - always ON</td>
</tr>
<tr>
<td>Multiple reference frames</td>
<td>Preset-dependent</td>
</tr>
<tr>
<td>HEVC B-frames</td>
<td>NVENCODE API</td>
</tr>
<tr>
<td>Others</td>
<td></td>
</tr>
</tbody>
</table>

- Higher throughput at same quality as Pascal
- Turing GPUs have single NVENC engine with higher quality
TURING NVENC QUALITY

- Focus on quality - RDO, multi-ref, HEVC B-frames, ...
- Quality vs performance trade-off
- Quality is content dependent
- 600+ videos of 10-20 secs each: Natural, animation, gaming, video conference, movies
- 720p, 1080p, 4K, 8K
- Quality: PSNR, SSIM, VMAF, subjective
- Perf: fps, number of 1080p streams per GPU
H.264 ENCODE BENCHMARK

Non latency critical - Turing vs Pascal vs x264

"iso" quality = x264 medium
# H.264 ENCODE BENCHMARK

Non latency critical - FFmpeg commands

<table>
<thead>
<tr>
<th>NVENC slow</th>
<th>-preset slow -bufsize BITRATE<em>2 -maxrate BITRATE</em>1.5 -profile:v high -bf 3 -b_ref_mode 2 -temporal-aq 1 -rc-lookahead 20 -vsync 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>x264 slow</td>
<td>-preset slow -tune psnr -vsync 0 -threads 4 -vsync 0</td>
</tr>
<tr>
<td>NVENC medium</td>
<td>-preset medium -rc vbr -profile:v high -bf 3 -b_ref_mode 2 -temporal-aq 1 -rc-lookahead 20 -vsync 0</td>
</tr>
<tr>
<td>x264 medium</td>
<td>-preset medium -tune psnr -threads 4 -vsync 0</td>
</tr>
<tr>
<td>NVENC fast</td>
<td>-preset fast -rc vbr -profile:v high -bf 3 -b_ref_mode 2 -temporal-aq 1 -rc-lookahead 20 -vsync 0</td>
</tr>
<tr>
<td>x264 fast</td>
<td>-preset fast -tune psnr -vsync 0 -threads 4 -vsync 0</td>
</tr>
</tbody>
</table>
HEVC ENCODE BENCHMARK

Non latency critical - Turing vs Pascal vs x265

Higher bitrate savings

Higher perf

"iso" quality = x265 medium
## HEVC ENCODE BENCHMARK

Non latency critical - FFmpeg commands

<table>
<thead>
<tr>
<th>Encoder Type</th>
<th>FFmpeg Command</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NVENC slow</strong></td>
<td><code>-preset slow -rc vbr_hq -b:v BITRATE -profile:v 4 -bf 2 -rc-lookahead 20 -g 250 -vsync 0</code></td>
</tr>
<tr>
<td><strong>x265 slow</strong></td>
<td><code>-preset slow -b:v BITRATE -bf 2 -tune psnr -threads 4 -vsync 0</code></td>
</tr>
<tr>
<td><strong>NVENC medium</strong></td>
<td><code>-preset medium -rc vbr_hq -b:v BITRATE -profile:v 4 -bf 2 -rc-lookahead 20 -g 250 -vsync 0</code></td>
</tr>
<tr>
<td><strong>x265 medium</strong></td>
<td><code>-preset medium -b:v BITRATE -bf 2 -tune psnr -threads 4 -vsync 0</code></td>
</tr>
<tr>
<td><strong>NVENC fast</strong></td>
<td><code>-preset fast -rc vbr_hq -b:v BITRATE -profile:v 4 -bf 2 -temporal-aq 1 -rc-lookahead 20 -g 250 -vsync 0</code></td>
</tr>
<tr>
<td><strong>x265 fast</strong></td>
<td><code>-preset fast -b:v BITRATE -bf 2 -tune psnr -threads 4 -vsync 0</code></td>
</tr>
</tbody>
</table>
RECONFIGURE DECODER

Video Codec SDK 8.2

No init time, reuse context, lowers memory fragmentation

✓ Input resolution
✓ Scaling resolution
✓ Cropping rectangle
✗ Codecs
✗ Bit-depth and chroma format
✗ Deinterlace mode
✗ Input resolution beyond max width or max height
DIRECT OUTPUT TO VIDMEM

Video Codec SDK 9.0

SDK 8.0 & earlier

CUDA pre-process ▶️ NVENC ▶️ CUDA Post-process

Video memory

Host/system memory

CPU process

PCle
OTHER UPDATES

- Video Codec SDK now supported on Power 9 + Tesla V100 SXM2
- High-level NVDEC error status
OPTICAL FLOW
New HW Functionality

- $4 \times 4$ optical flow vector, up to $4K \times 4K$
- Close to true motion
- Robust to intensity changes
- 10x faster than CPU; same quality
- New Optical Flow SDK
- Action recognition, object tracking, video inter/extrapolation, frame-rate upconversion
- Legacy ME-only mode support

TIPS FOR NVENC OPTIMIZATION
OPTIMIZATION STRATEGIES

General Guidelines

➢ Minimize PCIe transfers
  ➢ Eliminate, if possible
  ➢ Use CUDA for video pre-/post-processing
➢ Multiple threads/processes to balance enc/dec utilization
  ➢ Monitor using nvidia-smi: `nvidia-smi dmon -s uc -i <GPU_index>`
  ➢ Analyze using GPUView on Windows
➢ Minimize disk I/O
➢ Optimize encoder settings for quality/perf balance
FFMPEG VIDEO TRANSCODING

Tips

➢ Look at FFmpeg users’ guide in NVIDIA Video Codec SDK package
➢ Use `–hwaccel` keyword to keep entire transcode pipeline on GPU
➢ Run multiple 1:N transcode sessions to achieve M:N transcode at high perf
LOW LATENCY STREAMING (1/3)

Optimization tips

➢ Low latency ≠ Low *encoding* time

➢ Latency determined by
  ➢ B-frames
  ➢ Look-ahead
  ➢ VBV buffer size & avlbl bandwidth
LOW LATENCY STREAMING (2/3)

Optimization tips

➢ For 1-2 frame latency (e.g. cloud gaming), use
  ➢ RC_CBR_LOWDELAY_HQ & Low VBV buffer size
    ➢ Minimizes frame-to-frame variations
  ➢ Any preset (Default, HQ, HP preferred)
    ➢ LL presets have resolution-dependent behavior
  ➢ No look-ahead
  ➢ No B-frames
LOW LATENCY STREAMING (3/3)

Optimization tips

➢ Similar to HQ (non latency critical) encoding

➢ For higher (8-10 frames) latency (e.g. OTT, broadcast), use
  ➢ Any RC mode
  ➢ Any preset (default, HQ, HP preferred)
  ➢ VBV buffer size as per channel bandwidth constraints
  ➢ Look-ahead depth < tolerable latency
  ➢ B-frames as needed
VIDEO DL TRAINING

Typical Workflow

Loader → NVDEC → Augment → Decoded frames

Decoded frames → Color space → Resize → Crop → Reorder → Training

With DALI

- FFmpeg
- NVDECODE API
- CUDA post-processing

Instantiate operator:

```python
self.input = ops.VideoReader(device="gpu", filenames=data, sequence_length=len)
```

Use it in the DALI graph:

```python
frames = self.input(name="Reader")
output_frames = self.Crop(frames)
return output_frames
```
ROADMAP
ROADMAP
Video Codec SDK 9.1

➢ Q3 2018
➢ Error handling - Retrieve last error
➢ Perf/quality tuning
➢ Support for CUStream
RESOURCES


FFmpeg GIT: https://git.ffmpeg.org/ffmpeg.git

FFmpeg builds with hardware acceleration: http://ffmpeg.zeranoe.com/builds/

Video SDK support: video-devtech-support@nvidia.com

Video SDK forums: https://devtalk.nvidia.com/default/board/175/video-technologies/

Connect with Experts (CE9103): Wednesday, March 20, 2019, 3:00 pm